Proc. of the Int. Conference on Tools for Artificial Intelligence,
Washington, D.C., Nov. 6-9, 1990.

An Empirical Comparison of ID3 and HONNs for Distortion Invariant Object Recognition

Lilly Spirkovska and Max B. Reid

NASA Ames Research Center
M/S 244-4
Moffett Field, CA 94035-1000

Abstract

In recent comparisons of symbolic and neural learning algorithms, it has been shown that the ID3 symbolic learning algorithm performs better than a neural network trained using the backpropagation learning rule. However, none of the previous studies compared the performance of the two learning approaches for distortion invariant object recognition. Within this domain, we compared the ID3 system and a higher-order neural network (HONN). Our results show that HONNs are superior to ID3 with respect to recognition accuracy whereas, on a sequential machine, ID3 classifies examples faster once trained. A further advantage of HONNs is the small training set required. HONNs can be trained on just one view of each object, whereas ID3 needs an exhaustive training set.

Introduction

Both symbolic and neural network (connectionist) learning algorithms have been developed for the inductive acquisition of concepts from examples. The objective of both approaches is to learn which features determine the class of an object and use this knowledge to classify new objects accurately. The two approaches have been compared recently[1-3] on data sets typically used to test symbolic learning algorithms, such as soybean diseases and chess end games. However, none of the comparisons dealt with the performance of the two approaches on a data set neural network algorithms are typically applied to - object recognition.

This paper presents results of experiments comparing the performance of the ID3 symbolic learning algorithm[4] with a higher-order neural network (HONN)[5-7] in the distortion invariant object recognition domain. In this domain, the classification algorithm needs to be able to distinguish between two objects regardless of their position in the input field, their in-plane rotation, or their scale.

Background

ID3

The ID3 approach to pattern recognition and classification consists of a procedure for building an efficient decision tree from a set of training objects represented by attributes or feature values. At each node of the tree, the training objects are partitioned based on the value of the feature which provides the most information. The training set is recursively decomposed in this manner until the tree can correctly classify all the objects in the training set. A detailed description of the algorithm used to build a decision tree is presented in Quinlan[4].

Higher-order neural networks

A neural network is constructed of many simple elements called neurons connected with weighted arcs, as illustrated in Figure 1. As in the ID3 approach, the input to a neural network is a set of feature values. The learning process consists of adjusting the weights until the network is able to associate these feature values with the correct training object. Once trained, the network classifies novel examples by propagating the feature values of the new object through the network and summing the weighted products. Mathematically, the class membership decision is a function of the output node values, denoted by y_i for node i, which are calculated by:

$$y_i = \Theta(\Sigma_j\, w_{ij}\, x_j), \qquad (1)$$

where Θ is a nonlinear threshold function, the x_j's are the excitation values of the input nodes, and the interconnection matrix elements, w_{ij}, determine the weight that each input is given in the summation.

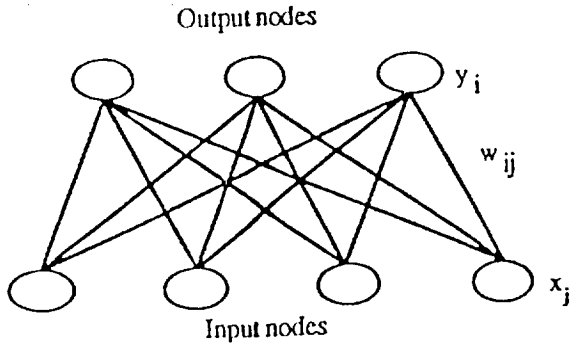

Figure 1: One layer of a neural network.

Note that the summation within the parenthesis in Eq. (1) is a function of the individual input pixels, x_j. No advantage is taken of any known relationships between the inputs. Thus, although multi-layer, first-order networks can learn translation, scale, and in-plane rotation invariances, they require a great deal of training, and produce solutions that are specific to particular training sets.

In contrast, higher-order neural networks (HONNs) are characterized by connections to neurons from combinations of inputs and can therefore have invariance information built into the network architecture. The output of a node in a general higher-order neural network is given by:

$$y_i = \Theta\left(\Sigma_j w_{ij} x_j + \Sigma_j \Sigma_k w_{ijk} x_j x_k + \Sigma_j \Sigma_k \Sigma_l w_{ijkl} x_j x_k x_l + \ldots\right). \quad (2)$$

A diagram of a neural network utilizing only third-order terms is shown in Figure 2.

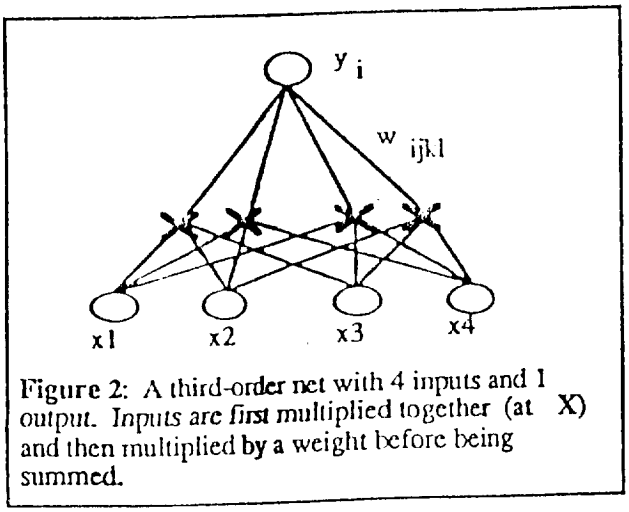

Figure 2: A third-order net with 4 inputs and 1 output. Inputs are first multiplied together (at X) and then multiplied by a weight before being summed.

A network with M inputs and one output using only rth order terms requires M-choose-r interconnections. For higher orders, this number, which is on the order of M^r, is clearly excessive. In order to overcome this combinatoric explosion of interconnections, invariances can be built directly into the network architecture by using information about relationships expected between the input pixels.[5-7] To achieve invariance to translation, scale, and in-plane rotation, a third-order network can be used. The output for a strictly third-order net is given by the function:

$$y_i = \Theta\left(\Sigma_j \Sigma_k \Sigma_l w_{ijkl} x_j x_k x_l\right). \quad (3)$$

Triplets of inputs form triangles with included angles (α, β, γ) as shown in Figure 3.

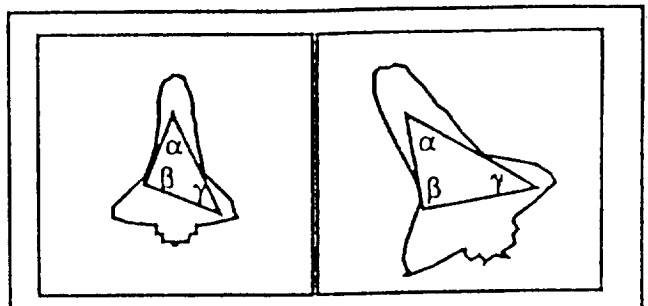

Figure 3: Because the included angles are invariant over scale, rotation, or translation, invariance is built into the network by using the same weight for all similar triangles formed by input triplets.

When the object is translated, scaled, or rotated in-plane, the three points in the same relative positions on the object still form the included angles (α, β, γ). In order to achieve invariance to all three distortions, all sets of triplets forming similar triangles are connected to the output with the same weight. That is, the weight for the triplet of inputs x_j, x_k, x_l is constrained to be a function of the associated included angles α, β, and γ:

$$w_{ijkl} = w_{i\alpha\beta\gamma} = w_{i\beta\gamma\alpha} = w_{i\gamma\alpha\beta}. \quad (4)$$

There are numerous advantages to this approach. First, the invariances are built directly into the network and require no learning to produce. Also, these invariances apply to any input pattern learned by the network. In contrast, it is believed that the hidden layers in multi-layered first-order networks represent the invariances. However, that is not usually obvious from looking at the final weights produced by the learning process. Finally, a HONN can perform nonlinear discrimination using only a single layer. Thus, rapid convergence can be achieved by using the simple learning rule:

$$\Delta w_{ijkl} = (t_i - y_i)\, x_j x_k x_l, \quad (5)$$

where t_i is the expected training output for node i and y_i is the actual output for node i.

Experimental Comparisons

Choice of algorithms

ID3 was chosen as a representative of the symbolic learning approach primarily for consistency with previous comparisons[1-3], in which it was studied because of its simplicity and popularity. It is the ancestor of several commercial rule induction systems and has been extensively tested on large data sets. Further, in experimental comparisons, ID3 generally performs as well or better than other symbolic learning algorithms[8].

To represent the neural network approach, we chose higher-order neural networks instead of the more popular backpropagation (backprop) trained network for several reasons. First, in previous studies[6,7], it was demonstrated that for distortion invariant pattern recognition, HONNs are superior to multi-layered first-order backpropagation trained networks in terms of training time, training set size, and accuracy. Second, in previous experiments[1-3], it was shown that backprop takes much longer to train than ID3. Because HONNs use a simpler learning rule than backprop, it was hypothesized that a HONN is more comparable to ID3. Finally, though HONNs compare quite favorably with other neural networks for distortion invariant object recognition, it is unknown how they perform compared with non-neural techniques.

Training sets

Both algorithms were trained using the T/C problem[9] and a simpler variation, the T/S problem. As described in Rumelhart, in the T/C problem, both characters are constructed of 5 squares, as illustrated in Figure 4, and the problem is to discriminate between them independent of translation or rotation. When these two patterns are considered over all translations and 90 degree rotations, configurations of triplets of squares must be examined to discriminate between them. If only distances between pairs of pixels are considered, the patterns are equivalent.

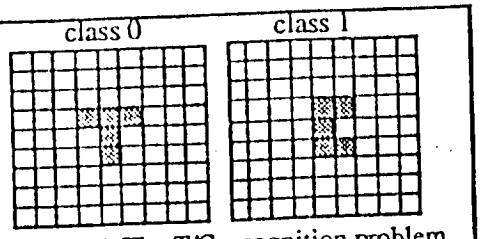

Figure 4: The T/C recognition problem, portrayed in a 9x9 pixel input field.

The T/S problem is similar except that there are only two unique 90 degree rotations possible with an S, as shown in Figure 5. Thus, there are only six possible rotated images as opposed to eight for the T/C problem.

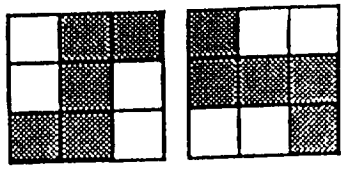

Figure 5: The two unique 90 degree rotations of an "S" in a 3x3 pixel input field.

The two algorithms were evaluated using three different training sets. Training set one was designed to determine the ability of the two algorithms to learn to distinguish between two objects regardless of an object's in-plane angular orientation of 0, 90, 180, or 270 degrees. The second training set was concerned only with whether the algorithms can learn to distinguish between two objects in a specified angular orientation regardless of changes in scale or position in the input field. Finally, the third training set combined the previous two training sets in order to determine the capabilities of the algorithms with respect to all three distortions - scale, position, or angular orientation - simultaneously.

As stated above, invariances can be built into the architecture of a HONN and a third-order neural network can learn to distinguish between two objects regardless of position, scale, or in-plane rotation based on just one view of each object. Therefore, the HONN algorithm was trained on just one view of each object. In contrast, in order for ID3 to guarantee 100% accuracy, it is necessary (though not sufficient) for it to be trained on an exhaustive set of views of the two objects. The number of images comprising each training set is cited below.

To evaluate the algorithms on only rotation invariance, case 1, it was trained on images drawn in a 3x3 pixel input field. The upright images, as in Figure 6a, were used for training the HONN and all versions of the images rotated by 90 degree increments were used for training ID3. For the T/C problem, as shown in Figure 6b, the training set for ID3 consisted of eight images, while for the T/S problem, it consisted of only six.

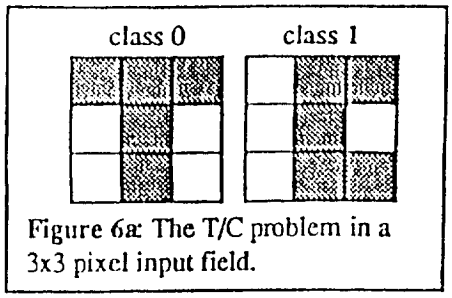

Figure 6a: The T/C problem in a 3x3 pixel input field.

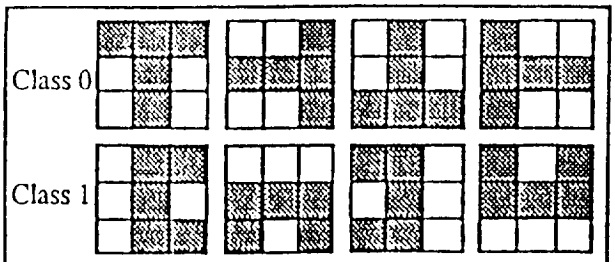

Figure 6b: The eight possible 90 degree rotated views of a T and a C.

For case 2, in order to demonstrate a reasonable amount of translation and scale, a 9x9 pixel input field was used, as in Figure 4. Again, only two images were necessary to train a HONN, while an exhaustive set was used for ID3. Considering all the possible positions for all three possible scales of the images, the training set consisted of 150 images for the T/C problem and of 132 images for the T/S problem.

To demonstrate all three distortions simultaneously, case 3, a 9x9 pixel input field was used. For the HONN algorithm, cases 2 and 3 are equivalent since all three distortions are built into the architecture. An exhaustive set for ID3 considering translation, scale, and rotation distortions consists of 600 images for the T/C problem and 396 images for the T/S problem.

Implementation details

Both algorithms were implemented in C, and a Sun 3/60 with 30 MB of swap space was used for simulations. The input pixels comprising the image were used as features and only binary images were considered, limiting the feature values to either 0 or 1. Thus, for the 3x3 pixel input field, there are 9 features, while for the 9x9 pixel field, there are 81 features.

The no-noise version of ID3 described in Quinlan[4] was used and neither ID3 nor the HONN algorithm was carefully optimized. Further, both algorithms were designed for 100% accuracy. In the results that follow, the statement that an algorithm was not able to learn to distinguish between two images means that it could not distinguish between them with 100% accuracy. Less accurate versions of the algorithms were not compared because the HONN is trained on just two objects. If the algorithm successfully learns to recognize an object, it is guaranteed to recognize all distorted versions of that object. If it cannot learn to recognize an object, testing distorted versions of that object is comparable to random guessing.

Results

Case 1: Rotation invariance only

The HONN learned to distinguish between a T and a C regardless of in-plane rotations in a 3x3 pixel input field in 55 training passes for a total training time of 2 seconds. For the T/S problem, it took 23 passes and only one second of training time.

For the T/C problem, ID3 could not learn to distinguish between the rotated versions of the two objects. Specifically, for the four images shown in Figure 7, the features do not provide sufficient information for ID3 to properly differentiate between the two classes. The feature, or pixel, values are shown in Table 1. Clearly, using the value of only one feature at a time is inadequate for deciding which class an object belongs to.

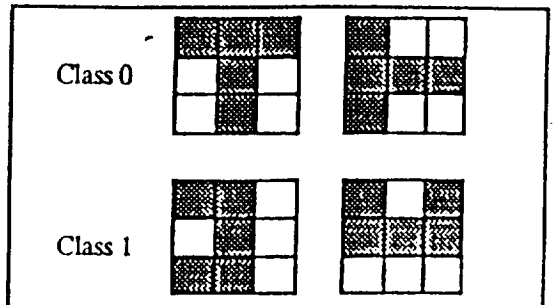

Figure 7: The features, as used by ID3, do not provide sufficient information to properly differentiate between the two classes.

Table 1: Feature values of the images in Figure 7.

pixel # 1 2 3 4 5 6 7 8 9	class
1 1 1 0 1 0 0 1 0	0
1 0 0 1 1 1 1 0 0	0
1 1 0 0 1 0 1 1 0	1
1 0 1 1 1 1 0 0 0	1

For the T/S problem, ID3 learned to distinguish between the rotated views of the characters in less than one second. Table 2 shows the feature values of the six training images. The resulting tree, shown in Figure 8, can be easily derived.

Table 2: Feature values of all the rotated views of a T and an S. Class 0 represents all T's and class 1 represents all S's.

pixel #	1 2 3 4 5 6 7 8 9	class
	1 1 1 0 1 0 0 1 0	0
	0 0 1 1 1 1 0 0 1	0
	0 1 0 0 1 0 1 1 1	0
	1 0 0 1 1 1 1 0 0	0
	0 1 1 0 1 0 1 1 0	1
	1 0 0 1 1 1 0 0 1	1

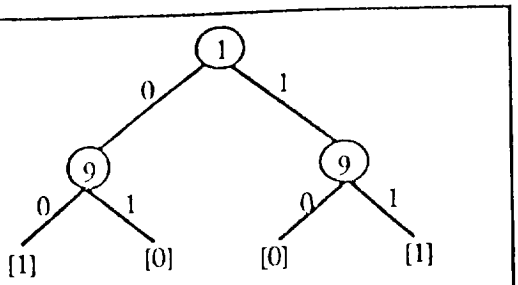

Figure 8: A decision tree constructed for the data in Table 2. The nodes (circled) represent the feature number, the leaves (bracketed) represent the class, and the branches are marked with the possible feature values.

Case 2: Translation and scale invariance

The HONN learned to distinguish between a T and a C regardless of translation, scale, or in-plane rotation in a 9x9 pixel input field in 55 passes and 57 seconds of training time. For the T/S problem, it learned to correctly classify all distorted views of the two images in 23 passes for a training time of 25 seconds.

ID3 created a decision tree in 30 seconds for the T/C problem and 33 seconds for the T/S problem.

Case 3: Simultaneous scale, translation and rotation invariance

As mentioned previously, case 2 and 3 are equivalent for the HONN algorithm since all three distortions are built into the architecture. Thus, the results for this case are the same as for case 2.

For the T/C problem, because ID3 was unable to learn to distinguish between the four images in Figure 7 and those same images are included in the exhaustive training set used in case 3, it was not tested on all three distortions simultaneously. For the T/S problem, though it was able to learn to distinguish between the two images regardless of their scale and position, or orientation, it was unable to learn the three distortions simultaneously.

Summary

Training set size

Because invariances were built directly into the architecture of the third-order neural network, it needed to be trained on only one view of each object. In contrast, in order for ID3 to guarantee 100% recognition accuracy, it was necessary, though not sufficient, for it to be trained on all views of each object. Thus, HONNs require many fewer images in the training set. This is useful if generating training and testing data is time consuming or otherwise difficult.

Recognition accuracy

For both the T/C and the T/S problems, the HONN was trained to distinguish between the two images regardless of their position, scale, or in-plane rotation in less than one minute. ID3, on the other hand, was unable to distinguish between either set completely. For the T/C problem, it could learn to distinguish between translated and scaled versions, but not rotated versions. For the T/S problem, it could learn all three distortions separately, but not simultaneously. In addition to the above results, HONNs have been extensively tested on numerous other objects, including a Space Shuttle Orbiter and an F-18 aircraft. A third-order network has learned all the examples presented to it within a small number of training passes (usually less than 50).

Training time

From the above results, it is difficult to compare the two approaches in terms of learning time because ID3 was unable to learn either problem completely. Also, for rotated versions, ID3 was trained for just 90 degree rotations whereas the HONN was designed to be automatically invariant to rotations by any angle.

In terms of testing, however, ID3 runs faster than a HONN. All the trees built for the above examples were fairly simple. The deepest tree had 60 levels and thus at most 60 comparisons. In a third-order net, triplets of inputs must be combined and then multiplied by their corresponding weight. For a 9x9 pixel input field, there are 81-choose-3 triplet combinations and thus 85,320 multiplications.

These results are valid only for implementations of the two algorithms on a uni-processor computer. The HONN algorithm learns (and subsequently uses) each weight independently of the rest. Therefore, a large speedup can be attained by running a parallel implementation on a multi-processor computer, such as the Connection Machine. However, since ID3 is based on the divide-and-conquer approach, it can also be parallelized to some degree. A fair comparison of the two algorithms can therefore only be based on the parallel version of both approaches.

Conclusions

Using a version of the T/C problem described in Rumelhart[9] expanded to include scale distortions, as well as a simpler variation of it, the T/S problem, we have shown that for recognition of objects regardless of their in-plane rotation, position in the input field, or scale, third-order neural networks are superior to ID3 in terms of recognition ability. Additionally, HONNs need to be trained on just one view of each object, whereas ID3 requires an exhaustive training set. Thus, if training data is difficult to obtrain, HONNs have an advantage over ID3. Finally, the results for training time were inconclusive because ID3 was unable to learn to distinguish between characters given all three distortions simultaneously. However, because ID3 creates very simple decision trees for the objects it learns successfully, it can classify examples faster than a HONN running on a sequential machine.

References

[1] R. Mooney, J. Shavlik, G. Towell, and A. Gove, "An Experimental Comparison of Symbolic and Connectionist Learning Algorithms," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August, 1989, pp. 775-780.

[2] D. H. Fisher and K.B. McKusick, "An Empirical Comparison of ID3 and Back-propagation," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August, 1989, pp. 788-793.

[3] S.M. Weiss and I. Kapouleas, "An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, August, 1989, pp. 781-787.

[4] J.R. Quinlan, "Induction of Decision Trees," *Machine Learning*, 1, 1986, pp. 81-106.

[5] G.L. Giles and T. Maxwell, "Learning, invariances, and generalization in high-order neural networks," Applied Optics, 26, 1987, pp. 4972-4978.

[6] M.B. Reid, L. Spirkovska, and E. Ochoa, "Simultaneous position, scale, and rotation invariant pattern classification using third-order neural networks," *Int. J. of Neural Networks*, 1, 1989, pp. 154-159.

[7] M.B. Reid, L. Spirkovska, and E. Ochoa, "Rapid Training of Higher-Order Neural Networks for Invariant Pattern Recognition," *Proc. Joint Int. Conf. on Neural Networks*, Washington, D.C., June 18-22, 1989, pp. 689-692.

[8] L.A. Rendell, H.H. Cho, and R. Seshu, "Improving the Design of Similarity-Based Rule-Learning Systems," *Int. J. of Expert Systems* 2, 1, 1989, pp. 97-133.

[9] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," *Parallel Distributed Processing*, 1, Ch. 8, MIT Press, 1986, pp. 348-352.